Bournemouth University

Faculty of Science and Technology

Department of Computing & Informatics

Submission Coursework Brief
Undergraduate

Unit: PROGRAMMING (COMP4015)		
Unit Contact: Ismail Alarab	**Credits:** 20	**Level:** 4

Assessment Title: Programming with Test Driven Development	
Assessment Number: 1 **of** 1	
Assessment Type: Individual	**Weighting:** 50%
Deadline: 22/11/2024 at 12:00 PM	**Submission Method:** Large File Submission Box (Code)

Quality Assessor (QA): Tim Orman	**Other Marker(s):** N/A

Can I use Generative AI tools?
Basic spelling and grammar correction tools are permitted.

The following originality requirements will apply to this assignment:

You are allowed to use any Generative AI or other AI powered tools, such as ChatGPT, for specific aspects as directed by the Unit Leader. Where any part of your assessment is sourced, or partially sourced from a generative AI tool, this requires a reference in the BU Harvard style.

Task:

This coursework consists of two types of tasks A and B.
Both tasks involve working with provided Python source code and must be completed. To receive marks for each task, the source code must function as per the requirements and must also pass the accompanying tests where appropriate. Upon completion of tasks A and B, you are required to submit the source code of each task and a documentation report (See appendix B).

TASK A: Assessed Lab Exercises [available on Brightspace]
You will demonstrate a professional approach to software development by systematically completing lab tasks. Some lab tasks during the first 8 weeks will be identified on Brightspace as ASSESSED TASK, with each lab task being available on Brightspace during the relevant week. Only the assessed lab exercises are to be submitted for assessment. You should do this by solving the given problems and testing your code to ensure that it meets the given requirements.
The tasks are designed to follow on from the concepts already covered in the lectures. You are required to test the input/output of your code and make sure that it meets the requirements of the task (*i.e. it works as required*). Before submission, all tasks from the lab exercises, each implemented as a separate Python function, must be combined into a single code file re-named as "**assessed_lab_exercises.py**" containing all the required functions.

TASK B. IMS Project Application

You are given a Python source code to download from this link and complete a simulated inventory management system (IMS) application. You are required to complete some of functions to make the code fully functional as required and pass the checks. The full detail of the IMS project application is in a separate document available on Brightspace and is also attached to this brief below (Appendix A).

Important Dates

Summary of dates & deadlines		
Event	**Expected Outcome**	**Deadline**
Start of Task A	This task will take place during lab sessions. All students are required to complete their assessed lab work during the lab time and save their work in their BU OneDrive folder. Once all assessed labs are completed, students must submit to Brightspace as instructed.	**Takes place between Friday 4th October - Friday 22nd November**
First Progress Review (GitHub Checks) – Task B	All students are required to: 1. Download the source code of Task B, 2. Create a new GitHub repository, 3. Push the original source code to the new GitHub repository with a meaningful commit message.	**Friday 25th October**
Start of Task B	You can start Task B as early as possible. Ideally, by this date, you should have all the necessary information covered in your lectures to start working on Task B.	**Friday 1st November**
Submission date of both tasks A and B	You are required to submit two things to Brightspace under "Assignment Submission" 1. Source code of Task A ("assessed_lab_exercises.py") and source code of Task B (.py files as provided) in a single zip file completed as required. 2. Coursework Report (PDF): A documentation code report (template in Appendix B) should document your contributed code in tasks A and B of any completed function. Further details on the report are given in next section below.	**Friday 22nd November 12pm**

A Code Documentation Report:

The code documentation report is a list of the functions you contributed to in Tasks A and B. You are provided with a code documentation report template (Appendix B) that can be downloaded from Brightspace. You are required to complete this template and submit it in PDF format.

For each function, include the following:

- The function name
- The input/output of the function
- An explanation of the function's purpose
- Any challenges you faced while working on the function or any existing issues in your code

This report allows you to demonstrate your understanding and reflect on your work.

In addition, provide screenshots of your private GitHub repository for Task B, including:

- The repository overview
- The commit history with meaningful commit messages
- Branches and pull requests (if applicable)

Include the GitHub link to your account and cite any references used in your coding.

Please note that the word count in the table documentation report should not exceed 500 words across all rows you fill in.

The report you are filling is considered supplementary material to Tasks A and B.

Any report is not accompanied by the source code will not be marked.

If you use generative AI in this coursework, then you should

1. Reference the tool used in BU Harvard Referencing format.
2. Attach a screenshot of the tool used showing your input (message prompt) and the output (even if you use part of it) in its entirety from the AI tool.

Intended Learning Outcomes (ILOs)

This unit assesses your ability to:

1. Design and implement solutions to different problems using appropriate programming languages.
2. Demonstrate the appropriate use of basic data-structures in software programs.
3. Demonstrate an understanding of the importance of developing software in an ethical, secure, and professional manner.
4. Test and debug a given program by using a suitable testing strategy.

Submission Format:

BEFORE the submission due date (see above) you should submit two things to Brightspace:

1. All source code files - in a ZIP file.
   Containing all submitted source code from Task A (assessed lab exercises) and Task B (IMS project application).
   All code should be: -
   - In the same project folder and structure, as it was written and tested, to ensure its compatibility while marking.
   - Compressed only ONCE ON THE OUTER MOST FOLDER with NO additional zipped folders/files contained within it.

    - In the correct format ie a .zip file, and NOT a rar file or any other compressed format.
2. Documentation report (in PDF format) - the form to be completed can be found at the end of this brief (see Appendix B) or on Brightspace, and details of what to include are above. Please complete the form and submit it in PDF format file.

As a safety measure, *there is no need to leave your submission until the last minute!*You can upload anytime from when the box opens up to the deadline.

How will this be assessed?

The following criteria will be used to assess the assignment: A guide to the levels of attainment (Appendix C) is attached to this brief.

Marks distribution

Criteria	ILO's	Marks
Criterion 1 **Task A (Assessed lab exercises)** Tasks to be completed and functional as required.	Weekly tasks completed as required (ILOs 1, 3 & 4)	30%
Criterion 2 **Task B (Project Application)** Code works and Passes Unit Tests i.e., functional code and conforms to requirements (25%) Good Code Practice, Quality, Understandability and Readability ○ Well selected data types, structures and algorithms (4%) ○ Comments (4%) ○ Layout and whitespace (4%) ○ Naming conventions, including meaning of name and the case used (4%) ○ Status of warnings and errors (4%)	Completed and tests pass (ILOs 1, 3 & 4) Quality of the code (ILOs 1, 2 and 3)	45%
Criterion 3 Code developed in a professional manner: ○ Code documentation report submitted (Report must be accompanied by the source code for both Tasks A and B) - (10%) ○ Submission completed and formatted as instructed (5%) ○ GitHub repository of Task B showing timely updates accompanied by appropriate commit messages (10%): ▪ Creating a new private GitHub repository and pushing the original source code files forTask B. ▪ Providing screenshots from your private GitHub repository (for Task B) and showing least three meaningful commit messages.	Code developed in a professional manner (ILO 2 & 4)	25%

Please, refer to Bournemouth University's Generic Assessment Criteria available on Brightspace that will guide you regarding the expected quality of your work.

Questions about the assessment:

Questions about the assignment brief can be asked during the lectures or labs. In addition, questions can be posted on the unit discussion board on Brightspace. This means ALL students have fair and equal access to the answers given. It will also be faster as questions can be answered by the first member of the unit team to be available. Emails should only be sent direct to the unit staff if they are of a private nature.

Academic Integrity

The work you submit must be your own. Any attempt to gain an unfair advantage in your assessment by **cheating**, deception or fraud is considered an academic offence. The 'Assessment help and support' section of the unit (found under 'Assessment' in the content area) provides more guidance on avoiding academic offences, including **any guidance on what will or will not be considered an academic offence in this specific assessment**

Help and support

The 'Assessment help and support' section of the unit (found under 'Assessment' in the content area) provides information and guidance, including specific information on support for this assessment. It provides help with our policies on deadline extensions and information on support available in the university, including academic skills support and additional learning support for students with disabilities.

Disclaimer

The information provided in this assignment brief is correct at time of publication. In the unlikely event that any changes are deemed necessary, they will be communicated clearly via e-mail and via the VLE and a new version of this coursework brief will be circulated.

Date Issued: 30/09/2024

Appendix A

Programming 2024 – Submission Coursework Task B

IMS Project Application (Project Description):

The goal is to develop an inventory management system (IMS) for Trikommerce, a simulated clothing store, to track and report the availability of T-shirt stocks at the end of each day. In particular, you will be required to complete some lines of coding of the provided Python source code to successfully develop the IMS. The system will ensure regular restocking of items to maintain stock levels and will automatically generate an Excel sheet (in CSV format) that shows daily periods, number of units sold, restocked units, and available units for each day over a specified period.

Upon successful completion, your system will be able to automatically generate a daily record in the form of a CSV file, detailing the daily total number of units sold, restocked, and available as shown in the table example below.

Day	Sold Units	Restocked Units	Available Units
0	0	2000	2000
1	105	0	1895
2	144	0	1751
3	47	0	1704
4	166	0	1538
5	43	0	1495
6	175	0	1320
7	0	680	2000
...	...	...	...
48	94	0	1729
49	0	271	2000

Table 1: Descriptive example of the content of a generated CSV report.

A manual managing of a store's inventory is challenging and can lead to overstocking of slow-moving items and stockouts of high-demand items. Therefore, you are required to implement the IMS following the requirements below to automate the process for this simulated clothing store, ensuring optimal stock levels and enhancing overall business efficiency.

Please note that this simulation is designed to fit the requirements of this coursework and may not reflect real-world scenarios.

Please read carefully requirements of IMS application (Sections (a) --> (e).) as follows.

Appendix A

Programming 2024 – Submission Coursework Task B

Requirements:

a) Initial considerations:

1. IMS reporting period: It starts on day 0 and ends on day 49 (Inclusive). Please note that the generated report has only one record for each day.
2. Restocking days: The first stocking begins on day 0 and occurs every seven days thereafter (i.e. days 7, 14, 21, …, 49). The initial stock level is set to 2000 units.
3. Selling days: Selling occurs only on days that are not-stocking days *(i.e. the number of sold units at any stocking day should be zero – Refer to Tables 1 and 2)*.
4. Stock limits: The number of restocked or available T-shirts should not exceed 2000 units on any day, even after restocking, to maintain optimal stock levels.
5. Sales limits: On selling days, the number of T-shirts sold daily is randomly assigned using Python's random function. The sales should not exceed 200 units on any selling day.

b) Project constraints:

1. The system should be written in Python programming language using Visual Studio Code (VSCode).
2. You are provided with the source code on **Brightspace** in the assessment folder (in the same area where you find this submission brief: https://brightspace.bournemouth.ac.uk/d2l/le/lessons/447604/lessons/2079917). You must only use and some lines of the given source code of files - as instructed in Section c - to ensure a fully functional IMS system.
3. The project should be uploaded to your GitHub account (create one if you do not have) in a new **GitHub Repository** and updated as appropriate to show all the events done and the progress of your project through **useful commit messages** where appropriate.
4. The IMS system should pass the provided unit tests as demonstrated in section (d).

c) Programming Instructions:

You are provided with **incomplete** source code (Figure 1) that has features with either partial or full functionalities. Upon running the given source code, you will see a main menu (Figure 2) with the following options:

1. Generate report (**Partially functional feature**): This option generates an empty excel sheet report of the IMS simulation.
2. Check report (**Fully functional feature**): This feature allows you to verify if the generated Excel sheet report meets the project requirements specified in Section (a) (i.e., it functions as required).
3. Exit (**Fully functional feature**): This option exits from the main menu.

Appendix A

Programming 2024 – Submission Coursework Task B

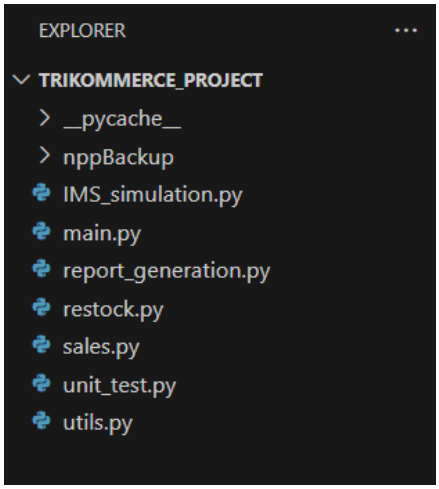

Figure 1. Python source code of files.

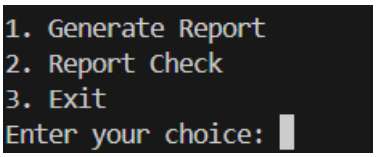

Figure 2. Main menu output of the source code.

Your final goal is to make the "**Generate Report**" feature fully functional according to the requirements. To do that, please follow all the requirements and complete the following modules (i.e., *.py files*) in your provided code:

- restock.py: Complete the function "**restock_inventory**". This function takes records of previous days - as list - and updates the stocking record for a current day following the requirements.
- sales.py: Complete the function "**daily_sales**". This function takes records of previous days - as list - and updates the sales for a current day following the requirements.

Function name	Function Input	Function Output	Function Description
restock_inventory	• available_items (int) • inventory_records (list) • current_day (int)	• available_items (int)	This function handles the restocking of inventory. On day 0 and every 7 days thereafter, it restocks the inventory to a maximum of 2000 units and updates **inventory_records** with the details.

Appendix A

Programming 2024 – Submission Coursework Task B

daily_sales	available_items (int), inventory_records (list), current_day (int)	available_items (int)	This function manages the daily sales of inventory. On non-restocking days, it generates a random number of sold units (up to 200) and updates **inventory_records** with the number of units sold and remaining inventory.

The final code output must be able to do the following:

i. **Generate report**: This feature should automatically generate an Excel sheet with records of daily stock levels. ***As an example***, a sample on a generated Excel sheet report for a couple of days is given in the example table below for your guidance. Please be aware that your code should be able to automatically generate a simulated sheet as required for the full required tracking period (*i.e., from 0 to 49 days*).

ii. **Check report**: You do not need to write any code for this feature. You need to use this feature in section (d).

iii. **Exit:**

Day	Sold Units	Restocked Units	Available Units
0	0	2000	2000
1	105	0	1895
2	144	0	1751
3	47	0	1704
4	166	0	1538
5	43	0	1495
6	175	0	1320
7	0	680	2000

Table 2: Example copied from an excel sheet report generated showing 2 stocking periods. The 1st stocking period is at day 0 and the 2nd stocking period is at day 7.

Appendix A

Programming 2024 – Submission Coursework Task B

d) Code Testing:

You are required to test your code to ensure that it is fully functional and pass all checks. You can do that by running the Python source code and using the "Check report" feature that appears in the main menu from the command-line interface (CLI)/ VSCode terminal. You can ensure that all the checks are passed from the returned message. This means that the requirements in section (a) are followed properly.

e) GitHub Upload and Records:

The IMS project application must be linked with a **private** GitHub repository during the project progress where all the project updates are recorded in GitHub and accompanied by appropriate and useful commit messages. To do so, follow these steps:

1. Create a private GitHub repository for the IMS project.
2. Link the IMS project application to this GitHub repository by uploading the original source code of files and including an appropriate commit message.
3. Regularly update the project in the repository as you make progress.
4. Include meaningful commit messages (at least three commits) with each update to clearly explain the changes made.

A summary of dates and deadlines table is provided in your submission brief above.

PROGRAMMING
Coursework Assessment Report (Appendix B)

Student name:_____ Seminar group: _____

I. Complete Table below:

Specify all functions you have worked on in Tasks A and B in the sub-brief and complete the table below. Complete the table below for each function by filling the cells of the table. Add as many rows as necessary to include all the functions you have contributed to. An example is provided below for your guidance.

Code Example:

```
def add_numbers(a, b):
    return a + b
```

Function name	What is the function input/output?	Explain the logic of the function.	How did you test it?	What are the challenges of this function?
add_numbers	Inputs: a, b (2 real numbers) Output: a+b (1 real number)	Adds two numbers and return the sum of them	Tests (positive, negative, mixed signs). (4, 3) → 7; (-4, 1) → -3; (-1,-2) → -3.	This function raised error when (a,b) are strings or complex numbers. Etc.

PROGRAMMING
Coursework Assessment Report (Appendix B)

II. Provide below screenshots to your GitHub repository and to your committed messages showing your progress properly.

---------- *Provide all screenshots properly showing your progress in GitHub here* ----------

III. Provide your GitHub link below:

----------*Your GitHub link*----------

IV. List all references, including AI-generated tools, if applicable. If you used AI-generated tools, specify the name of the tool, provide screenshots/texts of the input/output, and explain how it was used in your project submission.

PROGRAMMING
Coursework Assessment Report (Appendix B)

References:

[1].
[2].
[3].

Referencing Usage of AI-generated tools:

AI-generated tool	Input	Output	How is it used?

Guide to Levels of Attainment (Appendix C)

	Tasks Completed and Understood	Quality of Code	Professional Manner
Criteria examined	Tasks completed successfully i.e.: functional fulfilling task requirements and passing the tests where necessary	Well selected data types, structures and algorithms using appropriate methods. Layout and whitespace applied appropriately Naming conventions and meaningful names used Warnings eliminated	Comments applied appropriately Tasks completed in a timely fashion where appropriate with accurate and appropriate reflection on progress and use of AI, in report Submitted in correct format Uploaded to GitHub with useful commit messages
Excellent (> 80)	ALL or nearly all … tasks completed. Perhaps one is functional but not working as expected. High level of understanding demonstrated	No improvement (or very little) needed Selection of data types, structures algorithms and methods are sound on the whole. Regular insightful commenting. Good layout Good naming of variables. No warnings	No/Very little improvement needed Tasks completed in a timely fashion and insightful progress report. Submission completed as per brief
Good (65 – 80)	MOST… Most tasks completed Good level of understanding	Little improvement needed Selection of data types, structures algorithms and methods occasionally not best choice Regular commenting. Occasional layout error Good naming of variables – occasional error One warning	Little improvement needed Tasks completed in a timely fashion- perhaps one or two clearly not done in time - and considered progress report
Average (50 – 65)	SOME … Approx half tasks completed or perhaps 1) not much of the timed assessment and the lab tasks or 2) only the timed assessment completed. Occasional gaps in understanding	Some improvement needed Selection of data types, structures algorithms and methods considered although some not best choice. Layout mostly good. Occasionally naming variables lacks meaning/correct format. Some warnings dealt with.	Some improvement needed. Some useful comments Progress report shows gaps in timeline /missing - content contains only some reflection
Weak (40 – 50)	A FEW… Two tasks work as expected. Other code incomplete/does not compile Many gaps in understanding	Much improvement needed Selection of data types, structures algorithms and methods often not best choice Occasional comments. Poor layout in some places. Occasional mistakes in variable name format A few warnings dealt with.	Much improvement needed Progress report shows multiple gaps in timeline/missing - content contains little reflection
Failing (< 40)	NONE or VERY FEW… tasks completed /does not compile Little/no evidence of understanding	A lot of work needed to pass Poor selection of data types, structures algorithms and methods, if they are used. Poor naming of variables. Warnings wherever they normally occur still in place. No comments. Poor layout.	A lot of work needed to pass Progress report shows very little was done in a timely fashion and content contains very little/no reflection